Lattice QCD Archive Format

V1.02 October 25, 1998

1 Overview

This document describes the format of the data files stored in the QCD archive. Files consist of an ASCII header followed immediately by binary data. The header consists of any number of lines of the form $\langle keyword \rangle = \langle value \rangle$. Some of the keywords are required, or have default values. This is spelled out in sections 2 and 3. At this point, the binary data always consists of gauge links written in 32-bit IEEE big-endian format. Details are spelled out in section 4.

2 Header format

The header consists of some number of ASCII lines, beginning with a `BEGIN_HEADER`, and ending with `END_HEADER`. The lines in between are of the form $\langle keyword \rangle = \langle value \rangle$. Exactly one keyword should appearon each line. Certain keywords are required to appear, while others are optional. For example, we expect a 32-bit checksum of the binary data, but don't require that the creation date be specified. Some keywords require the presence of other keywords. For example, if the `DATATYPE` is `4D_SU3_GAUGE` (the only choice right now), then `DIMENSION_1` through `DIMENSION_4` are required. Some keywords have default values, and do not need to appear explicitly. For example, if `FLOATING_POINT` is not defined, data is assumed to be in the default 32-bit IEEE big-endian format.

The keywords are summarized in Table 1.

3 Keywords

Here are comments on each of the keywords defined to date.

- `DATATYPE`. A required keyword. Specifies the type of data. For now, the only defined value is "`4D_SU3_GAUGE`," which indicates a standard

Keyword	C Format	Sample Values	Comments
BEGIN_HEADER			required
HDR_VERSION	= %s	1.0	(default)
DATATYPE	= %s	4D_SU3_GAUGE	required
STORAGE_FORMAT	= %s	1.0	(default)
DIMENSION_1	= %d	24	required by 4D_SU3_GAUGE
DIMENSION_2	= %d	24	required by 4D_SU3_GAUGE
DIMENSION_3	= %d	24	required by 4D_SU3_GAUGE
DIMENSION_4	= %d	32	required by 4D_SU3_GAUGE
LINK_TRACE	= %f	0.8732148	required by 4D_SU3_GAUGE
PLAQUETTE	= %f	0.5771234	required by 4D_SU3_GAUGE
BOUNDARY_1	= %s	PERIODIC	(default) under 4D_SU3_GAUGE
BOUNDARY_2	= %s	PERIODIC	(default) under 4D_SU3_GAUGE
BOUNDARY_3	= %s	PERIODIC	(default) under 4D_SU3_GAUGE
BOUNDARY_4	= %s	PERIODIC	(default) under 4D_SU3_GAUGE
CHECKSUM	= %x	4f13a3c7	required
ENSEMBLE_ID	= %s	OSU_Q60a	required
SEQUENCE_NUMBER	= %d	10020000	required
ENSEMBLE_LABEL	= %s	Quenched b=6.0	optional
CREATOR	= %s	CU-QCDSP	optional
CREATOR_HARDWARE	= %s	QCDSP crate 13	optional
CREATION_DATE	= %s	Sun Jun 10 1990	optional
ARCHIVE_DATE	= %s	Mon Sep 28 1998	optional
FLOATING_POINT	= %s	IEEE32BIG	(default)
END_HEADER			required

Table 1: Keywords, their types as defined by the C format which writes them, and default or sample values.

4-dimensional hypercubical SU(3) gauge configuration. Some day one may want to archive eigenmodes of the Dirac operator in 7 dimensions, and define a value like "`7D_E8_EIGENMODE`." We imagine that associated with each datatype there will be a certain number of required keywords. For example, `4D_SU3_GAUGE` is implicitly hypercubic and requires four geometry parameters `DIMENSION_1` etc., plus a `PLAQUETTE`, while some future datatype might require a more geometry parameters, or a quark mass.

- `HDR_VERSION`. (Defaults to "1.0".) Specifies the version of this header. This may stay at version 1.0 forever, but is here just in case.

- `STORAGE_FORMAT`. (Defaults to "1.0".) Specifies the version format for this datatype. Again, this may stay at version 1.0 forever. For `4D_SU3_GAUGE`, version 1.0 implies various defaults, e.g. 2 rows of IEEE32 numbers in the order described below.

- `DIMENSION_1` etc. (Required by `4D_SU3_GAUGE`.) Specifies the hypercubical geometry. `DIMENSION_1` is the fastest moving direction; `DIMENSION_4` is the slowest. For the record, we think of 1 as the $x-$direction, $2 \leftrightarrow y$, $3 \leftrightarrow z$ and $4 \leftrightarrow t$.

- `BOUNDARY_1` etc. (Optional under `4D_SU3_GAUGE`.) Specifies the boundary conditions in one of the directions. Defaults to `PERIODIC`.

- `CHECKSUM`. Required by all. The checksum is an unsigned, 32-bit integer sum of all the 32-bit words making up the binary block following the header. The ordering is big-endian, so on little-endian machines, one has to do the byte transposition before computing the checksum. Getting this right gives reassurance that the bytes are correctly assembled in to words.

- `LINK_TRACE`. Required under `4D_SU3_GAUGE`. The average trace of a link, normalized to 1.0 in the trivial configuration, i.e.

$$\frac{1}{12N_{\rm site}}\sum_{n,\,\mu} {\rm Re\ Tr}\ U_\mu(n) \tag{1}$$

  Note that the trace is over all three colors, and includes one reconstructed element. Getting this right gives reassurance that that the data has been assembled in to links correctly.

- PLAQUETTE. Required under 4D_SU3_GAUGE. The average trace of the plaquettes, normalized to 1.0 in the trivial configuration, i.e.

$$\frac{1}{18N_{\mathrm{site}}}\sum_{n,\,(\mu>\nu)} \mathrm{Re\ Tr}\ \{U_\mu(n)U_\nu(n+\hat{\mu})U_\nu(n+\hat{\nu})^\dagger U_\nu(n)^\dagger\} \qquad (2)$$

- ENSEMBLE_ID. Required by all. Unique string to identify the ensemble. We expect this to be something like "QCDSP_DW98", where the first view characters identify the contributing collaboration, and the last few are a run number. This string forms part of the file name, so we don't want it to be too long.

- SEQUENCE_NUMBER. Required by all. A unique integer identifying the configuration within its ensemble. Typically, this will be a trajectory or sweep number. For the convenience of the end user, we want all sequence_numbers within an ensemble to have the same number of significant digits. Therefore we may offset the numbers by some large power of 10. E.g. given an ensemble of configurations from trajectory 200 to trajectory 30000, we might number them 1000200 to 1030000.

- ENSEMBLE_LABEL. An optional string giving a short description of the ensemble, e.g. "Dynamical Staggered 8**4 beta=5.4 m=0.01". Given the large number of parameters which may be needed—e.g. β, β_A, κ, C_{SW}, algorithm, stepsize, trajectory length, etc.—we do not expect this line will fully describe an ensemble. This field is meant to provide a quick description, while details will be given on the ensemble's webpage.

- CREATOR. An optional string identifying the collaboration or individual who provided the data.

- CREATOR_HARDWARE. An optional string describing the computer on which the data was generated. Sample values: "QCDSP crate 13" or "mcurie.nersc.gov".

- CREATION_DATE. Optional string, such as that generated by the Unix command "date," specifying the date the data was generated.

- ARCHIVE_DATE. Optional string specifying the date the data was written in archive format.

- FLOATING_POINT. Optional string specifying the floating point format of the binary data. By default we use 32-bit IEEE in big-endian order (`IEEE32BIG`), but we leave the option open for future needs, e.g. 64-bit precision. `IEEE32` is a synonym for `IEEE32BIG`.
- Ensemble-dependent keywords. A given ensemble may define other keywords appropriate for its action. For example, in the first wave of ensembles we have defined `BETA` for the plaquette action and `QUARK_MASS` for dynamical ensembles. Future ensembles may require a longer list of parameters, which could be specified in this way.

4 Storage Order (for 4D_SU3_GAUGE)

As indicated above, immediately following the header comes the binary data. At present we are archiving only 4-dimensional SU(3) gauge configurations (`4D_SU3_GAUGE`). For these, the basic object is the link matrix $U_\mu(n)$ which implements parallel transport to the site n from the site one step in the positive μth direction $(n+\hat{\mu})$. That is, if $\chi(n)$ transforms as a color triplet at the site n, then the combination

$$\chi^\dagger(n)U_\mu(n)\chi(n+\hat{\mu}) = \sum_{i,j}[\chi^\dagger(n)]_i[U_\mu(n)]_{ij}[\chi(n+\hat{\mu})]_j \tag{3}$$

is gauge invariant.

In terms of the indices i, j defined just above, we store a link as complex numbers in the order $\mathrm{Re}\,[U_{11}]$, $\mathrm{Im}\,[U_{11}]$ followed by $U_{12}, U_{13}, U_{21}, U_{22}, U_{23}$. By default, we store only these two rows, understanding that the third row can be rescontructed as the cross-product. If there is need in the future, we will define a keyword (with default value 12) to indicate configurations where we store the full 18 word format.

We store the four links $U_\mu(n)$ adjacently in the order $\mu = 1, \cdots, 4$. Of the spacetime indices, $DIMENSION\_1$ moves the fastest, followed by 2, 3 and 4. That is, with indices arranged according to C convention we store the data as an array

$$U[t][z][y][x][\mu][i][j] \in U[NT][NZ][NY][NX][4][2][3] \tag{4}$$

while in Fortran we would create

$$\text{complex } U(3,2,4,NX,NY,NZ,NT) \tag{5}$$